GEORGIA INSTITUTE OF TECHNOLOGY — OFFICE OF CONTRACT ADMINISTRATION

PROJECT ADMINISTRATION DATA SHEET

[X] ORIGINAL [ ] REVISION NO. _____

Project No. A-3733 GTRI/~~GIT~~ DATE 1 / 9 / 84

Project Director: Jeff Hopper ~~School~~/Lab ECSL/CTAD

Sponsor: Scientific-Atlanta, Inc.

Type Agreement: P. O. No. 081980

Award Period: From 12/29/83 To 2/28/84 (Performance) 2/28/84 (Reports)

Sponsor Amount:	This Change	Total to Date
Estimated:	$ 12,237	$ 12,237
Funded:	$ 12,237	$ 12,237

Cost Sharing Amount: $_____ Cost Sharing No: _____

Title: Design and Testing of Firmware For the DS-3 Multiplexor

ADMINISTRATIVE DATA — OCA Contact Brian J. Lindberg X4820

1) Sponsor Technical Contact:

Mr. Robin Larson 925-5289
Scientific-Atlanta
Instrumentation Department
P. O. Box 105027
Mail Stop 28-I
Atlanta, GA 30348

2) Sponsor Admin/Contractual Matters:

Linn Parkinson
Scientific-Atlanta
Purchasing Department
4388 Shackleford Road
Norcross, GA

Defense Priority Rating: N/A Military Security Classification: N/A

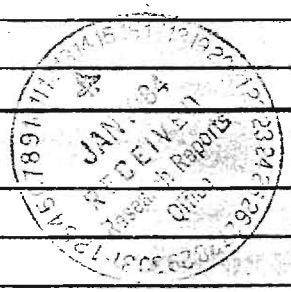
(or) Company/Industrial Proprietary: N/A

RESTRICTIONS

See Attached N/A Supplemental Information Sheet for Additional Requirements.

Travel: Foreign travel must have prior approval — Contact OCA in each case. Domestic travel requires sponsor approval where total will exceed greater of $500 or 125% of approved proposal budget category.

Equipment: Title vests with Sponsor; however none proposed.



COMMENTS:

COPIES TO:

Project Director (Hopper)
Research Administrative Network
Research Property Management
Accounting
Procurement/EES Supply Services
Research Security Services
(Reports Coordinator (OCA))
Research Communications (2)
GTRI
Library
Project File
Other I. Newton

GEORGIA INSTITUTE OF TECHNOLOGY OFFICE OF CONTRACT ADMINISTRATION

SPONSORED PROJECT TERMINATION/CLOSEOUT SHEET

Date April 10, 1984

Project No. A-3733 ~~School~~/Lab ECSL/CTAD

Includes Subproject No.(s)

Project Director(s) Jeff Hopper GTRI / ~~Other~~

Sponsor Scientific-Atlanta, Inc.

Title Design and Testing of Firmware for the DS-3 Multiplexor

Effective Completion Date: 3/15/84 (Performance) 3/15/84 (Reports)

Grant/Contract Closeout Actions Remaining:

- [ ] None
- [x] Final Invoice or Final Fiscal Report
- [ ] Closing Documents
- [ ] Final Report of Inventions
- [ ] Govt. Property Inventory & Related Certificate
- [ ] Classified Material Certificate
- [ ] Other

Continues Project No. Continued by Project No.

COPIES TO:

Project Director
Research Administrative Network
Research Property Management
Accounting
Procurement/EES Supply Services
Research Security Services
Reports Coordinator (OCA)
Legal Services

Library
GTRI
Research Communications (2)
Project File
Other

Form OCA 60:1028

Georgia Institute of Technology
ENGINEERING EXPERIMENT STATION
Atlanta, Georgia 30332

January 31, 1984

Linn Parkinson
Scientific-Atlanta, Inc.
Purchasing Department
4388 Shackleford Road
Norcross, Georgia 30348

Subject: Monthly Progress Report for the Period of December 29, 1983 to January 31, 1984 for Project A-3733-000

Dear Madam/Sir:

The firmware for Scientific-Atlanta's new 4641 Scanning D-3 switch is on schedule. After the specifications were determined at the beginning of this month, the software was designed and written. Approximately 90% of that software has been entered into our microprocessor development system (MDS). What remains is modification of existing communication routines, details which will arise when all of the software is linked together and any changes which might have risen since the original design.

Although some testing of the software can be performed on the MDS, a prototype of the 4641 Scanning Switch will be needed in the next two weeks for system integration and testing, if we are to remain on schedule. There will be further communication concerning when a prototype will be available as that time approaches. If there are any questions or comments, please call me or Juan Santamaria at 894-3456.

Sincerely,

Mr. Jeffrey C. Hopper
Research Engineer I
Computer Technology and
Applications Division
Electronics and Computer
Systems Laboratory

JCH/jm

cc: Robin Larson (S-A), Cain, Williams, Santamaria, Project A-3733 Correspondence and Contract Files

ENGINEERING EXPERIMENT STATION
GEORGIA INSTITUTE OF TECHNOLOGY
ATLANTA, GEORGIA 30332

4641 DS-3 SCANNING SWITCH SOFTWARE

FINAL REPORT FOR PROJECT A-3733-000

MARCH 1984

SUBMITTED TO:

SCIENTIFIC-ATLANTA, INC.

SUBMITTED BY:

COMPUTER TECHNOLOGY AND APPLICATIONS DIVISION
ELECTRONICS AND COMPUTER SYSTEMS LABORATORY
ENGINEERING EXPERIMENT STATION

TABLE OF CONTENTS

		Page
I.	INTRODUCTION	1
II.	SPECIFICATIONS	1
	A. GENERAL	2
	B. LOCAL OPERATIONS	3
	C. REMOTE OPERATIONS	4
III.	INTERFACE DEFINITIONS	6
IV.	SOFTWARE DESCRIPTION	9

APPENDIX I - PSEUDO CODE FOR PRIMARY MODULES

APPENDIX II - DATA BASE DESCRIPTION

I. INTRODUCTION

The Scientific-Atlanta 4641 DS-3 Scanning Switch is designed to allow time-multiplexed monitoring of several communication channels using only one bit error rate test receiver. The 4641 will accept front panel or remote interface commands that will configure the system to monitor the desired input. Each 4641 can accept 16 inputs and has the capability to drive 15 other slave units. The master unit has LED displays that indicate the current channel for a selected unit. The Engineering Experiment Station of Georgia Tech was contracted by Scientific-Atlanta to design, write, and test the 4641 software. This document describes the firmware required to perform the functions of the 4641 Scanning Switch.

II. SPECIFICATIONS

This section will first give a brief description of the hardware (supplied by Scientific-Atlanta) followed by a list of software specifications for local and remote operations.

The master unit has 16 input connectors and one output connector. Relays are programmed to switch one input to the output. The master has unit and channel rocker switches which will allow the operator to select relays in the master as well as slave units. There are seven segment displays on the master that show the unit number and its corresponding channel. There is also a single LED which indicates whether the unit is being remotely controlled via RS-232 or IEEE-488 interface.

The slave units, like the master, each have 16 input connectors and one output connector. The master sets the slave to specified channels us-

ing a parallel bus. Indication of whether an addressed unit is present in a particular configuration is available through this bus. There are no select switches or displays in the slave units.

The master unit uses an 8080 microprocessor to read input switches, display unit and channel numbers, select relays, communicate with slave units, and accept remote commands. Most interfaces with the processor involve simply reading or setting TTL latches. The seven segment displays are driven by BCD decoders. Either RS-232 or IEEE-488 interfaces (not both) can be present in the system. An 8251A USART handles the serial communication and a Motorola 68488 chip services the GPIB. Power-fail and remote service are the only interrupts provided. A dead-man reset is provided to insure that the processor will not stay in an undesirable state. The unit has provisions for 8K bytes of ROM and 2K bytes of non-volatile RAM.

The specifications of the 4641 software are listed below.

A. General

1. Keep current configuration (units in system and their selected channel) in NV RAM.
2. Perform CRC algorithm on NV RAM on power-fail to determine if units must be refreshed or initialized at power-up.
3. Continuous monitoring of units present in the configuration.
4. In the pristine state, all units (that are present in the configuration) are set to channel 1.
5. If a unit is added while the system is running, channel 1 is selected for that unit.

6. There are sixteen possible units (including the master) in a configuration. They have address designations of 0 to F(Hex). Unit 1 is reserved for the master. Slaves have switches which can be set to any of the other available addresses.
7. Channels are designated from 01 to 16.
8. Remote operations are disabled on power-up.

B. Local Operations

1. The unit rocker switch is used to change the unit display and view the corresponding channel. It can either increment or decrement depending on the direction the switch is pushed. If the button is held in one position, the unit number will auto-increment or auto-decrement. If the unit that is selected is not present in the system, the channel number is blanked. If units "A," "B," "C," "D," "E," "F," or "0" are not present, an increment from unit 9 will result in a unit 1 display and a decrement from unit 1 will result in a unit 9 display. Otherwise, incrementing and decrementing will result in a normal procession (with wrap-around) through all 16 unit numbers. The current displays will blank on hex digits A, B, C, D, E, and F. Still, the correct digit is output to the encoder.
2. The channel rocker switch is used in the same manner as the unit switch. Each change results in an update of the relays in the selected unit to the new channel. Auto-repeat and wrap-around are also implemented for the channel switch.

3. The channel display will be blanked if the unit is not present by outputting an "AB" hex. If full hex encoders are later used, this could be interpreted as "Absent Box."

4. No front panel switches will be read while remote enabled.

C. Remote Operations

1. Valid Commands:

   "SWRE<t>" - Remote Enable

   "SWRD<t>" - Remote Disable (Same as Go To Local)

   "SWGL<t>" - Go To Local (Same as Remote Disable)

   "SWE1<t>" - Enable Echo of RS-232 Characters

   "SWE2<t>" - Disable Echo of RS-232 Characters

   "SWQS<t>" - Status Query

   "SWQC<t>" - Configuration Query

   "SWU$n_1n_2n_3$<t>" - "Who Are You" Query

   "SWCL<t>" - Device Clear

   <t> = Terminator = comma, semicolon, carriage return, or line feed

2. All responses to querys are terminated with ETX, CR, LF.

3. The "Who Are You" query output is a standard S-A format that gives hardware model and firmware revision number.

4. The status query output is a single ASCII character that is the addition of the contents of the status register and the letter "A." The status register has the following format:

   Bit 0 = Command Error

   Bit 1 = Power-Fail (CRC Passed on Power-Up)

Bit 2 = Tried to Access Unit Not in System Through "U" Command

Bit 3-7 = 0

The status register will be reset to 0 after the query status has been made.

5. The configuration query outputs the channel selection for all units in the system. It is a two line format shown below:

   03 12 05 01 XX 14 16 11

   01 XX 03 02 10 15 05 07

   The first line indicates the channels selected for units 1 through 8. The second line indicates units 9, A, B, C, D, E, F, and 0 in that order. If a unit is not present, an "XX" appears in that position. If all the units in the second line are not present, the entire line is not output.

6. The unit select is of the form: SWU$n_1n_2n_3$ where n_1 = unit number (0 through F Hex) and n_2n_3 = channel selection (01 through 16 decimal).

7. The RS-232 echo is initially enabled.

8. XON-XOFF protocol is obeyed.

9. In addition to the listed commands, the system will also respond to IEEE-488 specific commands of Device Clear, Go To Local, and Remote Enable.

10. A device clear will cause all units to be changed to channel 1.

11. When in local mode, the only command that will be processed is the remote enable command.

12. A command error will occur only if the bad command started with "SW" and the unit is remote enabled.

13. If a command error occurs, the decimal representation of the first letter after the "SW" is displayed on the unit and channel number displays for a short period.

III. INTERFACE DEFINITIONS

The hardware and software interface definitions are discussed in this section.

Memory:

ROM: 0000H - 1FFFH (8K)

RAM: 4000H - 47FFH (2K NV)

I/O: (Bit 0 = LSB, Bit 7 = MSB)

Address	I/O	Description
A0H	0	Reset Dead Man Timer
60H	0	Channel Number Display
		Bits 0-3 = Least Significant Channel BCD Digit
		Bits 4-7 = Most Significant Channel BCD Digit
68H	0	Unit Number Display and Control Bits
		Bits 0-3 = Unit Number
		Bit 4 = Lamp Test (Active Low)
		Bit 5 = Blank LEDs (Active Low)
		Bit 6 = Remote LED (Active Low)
		Bit 7 = Slave Unit Strobe (Active Low)
78H	0	Reset Rocker Switch Latch

Address	I/O	Description
00H	I	Front Panel Switches (Active High)
		Bit 0 = Increment Unit
		Bit 1 = Decrement Unit
		Bit 2 = Increment Channel
		Bit 3 = Decrement Channel
		Bit 4-7 = Not Used
80H	I	Status
		Bit 0 = Selected Slave Present (Active Low)
		Bit 1-5 = Not Used
		Bit 6 = Power Fail (Active High)
		Bit 7 = Not Used
08H	O	Remote Unit Output
		Bit 0-3 = Remote Channel Number
		Bit 4-7 = Remote Unit Number
10H	O	Master Unit Channel 9-16 Select (Active High)
		Bit 0 = Relay 9 On
		Bit 1 = Relay 10 On
		Bit 2 = Relay 11 On
		Bit 3 = Relay 12 On
		Bit 4 = Relay 13 On
		Bit 5 = Relay 14 On
		Bit 6 = Relay 15 On
		Bit 7 = Relay 16 On

Address	I/O	Description
18H	O	Master Unit Channel 1-8 Select (Active High)
		Bit 0 = Relay 1 On
		Bit 1 = Relay 2 On
		Bit 2 = Relay 3 On
		Bit 3 = Relay 4 On
		Bit 4 = Relay 5 On
		Bit 5 = Relay 6 On
		Bit 6 = Relay 7 On
		Bit 7 = Relay 8 On
20H-27H	I/O	68488 Registers
40H	I/O	8251 Data Register
44H	I/O	8251 Command Register
50H	O	Clock Pulse for 8253 Counter 2
58H	I	Baud Rate Select
		Bit 0-4 = Not Used
		Bit 5-7 = Baud Switches
		000 = 110 Baud
		001 = 300 Baud
		010 = 600 Baud
		011 = 1200 Baud
		100 = 2400 Baud
		101 = 4800 Baud
		110 = 9600 Baud
		111 = 19200 Baud
48H	I/O	8253 Counter 0 (Baud Rate)
4AH	I/O	8253 Counter 1

Address	I/O	Description
4CH	I/O	8253 Counter 2
4EH	I/O	8253 Mode Register

Remote units are updated by first outputting the proper unit and channel number to Port 08H. Then if Bit 0 of Port 80H is low, the strobe (Bit 7 of Port 68H) is first output to low and then back to high.

IV. SOFTWARE DESCRIPTION

The software can be divided into two major parts: Main Line and Interrupt Service. The Main Line consists of initialization and updating of relays and displays based upon front panel operation. If the unit is remote enabled, the Main Line handles the LED display of remote error conditions. After initialization, the Main Line samples the switches at approximately a 50 msec rate. Auto-repeat delays and error displays are based on this time.

The interrupt service must handle power down conditions and remote operations. Decoding of commands, queueing and outputting messages, and updating relays based upon remote entry are all done in the context of interrupt service.

The following list is a summary of the primary modules that make up the 4641 software.

"INIT" - Initialization of 4641 and 50 msec timing loop.

"MAIN" - Main line, executed approximately every 50 msec. Reads switches and updates relays and displays.

"INTR" - Interrupt service routine.

"MEMCHK" - Non-volatile RAM check.

"UPDATE" - Updates one specific unit and LED displays.

"MSTRUP" - Updates master unit.

"RERR" - Decodes and displays remote command error information

"NEWCHK" - Determines which units are present.

"RXCHAR" - Remote command message interpreter.

"RINTR" - RS-232 Interrupt Service Routine.

"GINTR" - IEEE-488 Interrupt Service Routine.

"GETBYT" - Determines next byte to be transmitted to remote device.

"QUEUP" - Queues messages for remote input.

These modules all have pseudo-code documentation. A list of utility routines and their description appear below:

"GINIT" - IEEE-488 Initialization called by INIT.

"RINIT" - RS-232 Initialization called by INIT.

"BINIT" - Message buffer and queuing initialization called by INIT.

"CRCC" - CRC calculation routine called by MEMCHK.

"CRCINT" - CRC initialization called by MEMCHK.

"SET9" - Determines if any of units A, B, C, D, E, F, or 0 are present. Sets variable CHAN16 to 17 if one is present. Otherwise CHAN16 = 10.

"FIRSET" - Takes channel values of units in memory and updates all units. Calls UPDATE and SET9. Called by STCHN1.

"STCHN1" - Sets all units to channel 1 memory and then updates the units, calls FIRSET. Called by INIT, RXCHR, and GINTR.

"COMPAR" - Utility compare routine called by INIT and RXCHAR.

"ASCBCD" - Converts ASCII number to 3 BCD digits. Called by RERR.

"NIBBLE" - Converts ASCII 0-F into Hex nibble. Called by RXCHAR.

"HEXBCD" - Converts channel number into ASCII representation. Called by RXCHAR.

The pseudo-code documentation for each primary module and a description of the data base are appended to this document.

APPENDIX I

PSEUDO CODE FOR PRIMARY MODULES

PROCEDURE INIT

OVERVIEW :

This routine initializes the instrument and then goes into a timing loop that causes MAIN to be called approximately every 50 msec.

INPUTS : None

OUTPUTS : Initially sets relays and displays.

CALLED BY : Executed on reset

CALLS ROUTINES : GINIT, RINIT, BINIT, STCHN1, FIRSET, MAIN, MEMCHK

PSEUDO CODE :

```
Initialize stack
CALL Ginit
CALL Rinit
CALL Binit
Start lamp test
CALL Memchk
IF New CRC value = Old CRC Value THEN
  Set Power fail bit in status register
  CALL Firset
ELSE
  CALL Stchn1
  Reset Power fail bit in status register
ENDIF
Initialize variables and flags
Enable interrupts
LOOP Forever
  Delay 50 ms.
  CALL Main
ENDLOOP Forever
```

```
PROCEDURE  MAIN

OVERVIEW :
     This routine reads front panel switches and updates units and
     displays. In remote operation, it outputs error conditions to
     the display.

INPUTS :   Front panel switches

OUTPUTS :  Displays and remote unit relay positions

CALLED BY :  INIT

CALLS ROUTINES : UPDATE, NEWCHK, SET9, RERR

PSEUDO CODE :

     Reset Dead-man timer
     IF Lamp test timer is not zero THEN
       Decrement timer
       IF Timer is zero THEN
         End Lamp Test
       ENDIF
     ELSE
       CALL Newchk
       CALL Set9
       IF Remote enabled THEN
         CALL Rerr
       ELSE
         Read front panel switches
         IF Switch reading is the same THEN
           Decrement Repeat flag
           IF Repeat flag positive THEN
             Exit
           ELSE
             Set repeat flag to short delay
           ENDIF
         ELSE
           Set repeat flag to long delay
         ENDIF
         IF Increment unit switch is high THEN
           IF Unit number = Chan16 THEN
             Unit number = 0
           ENDIF
           Increment Unit Number
         ELSE
           IF Decrement Unit Switch is high THEN
             IF Unit number = 1 THEN
               Unit number = Chan16
             ENDIF
             Decrement unit number
           ENDIF
         ENDIF
         Access the proper channel number indexed by unit number
         IF Increment channel switch is high THEN
           IF Channel number = OFH THEN
             Channel number = -1
           ENDIF
           Increment channel number
         ELSE
           IF Decrement channel switch is high THEN
```

PROCEDURE INTR

OVERVIEW :
This routine determines the cause of an interrupt and takes the appropriate action.

INPUTS : None

OUTPUTS : None

CALLED BY : Executed on interrupt

CALLS ROUTINES : MEMCHK, RINTR, GINTR

PSEUDO CODE :

```
Save registers
IF Power fail THEN
  CALL Memchk
  Halt
ELSE
  IF IEEE-488 Interface present THEN
    CALL Gintr
  ENDIF
  IF RS-232 Interface present THEN
    CALL Rintr
  ENDIF
ENDIF
Restore registers
Enable interrupts
```

PROCEDURE MEMCHK

OVERVIEW :
     This routine exercises a cyclic redundancy check on a portion
     of non volatile ram and places the results in CRCSAV.

INPUTS : None

OUTPUTS : Two byte result in CRCSAV

CALLED BY : INIT, INTR

CALLS ROUTINES : CRCINT, CRCC

PSEUDO CODE

```
     CALL Crcint
     FOR  I = 1 to 17
      Extract UNR-1 + I
      CALL Crcc
     NEXT I
```

PROCEDURE UPDATE

OVERVIEW :
This routine updates the unit whose address is in the unit number register (UNR) with trhe channel information found in the 16 byte array, UBUFF. It also updates the unit and channel LED displays.

INPUT : UNR and UBUFF

OUTPUT : Relay control to selected unit and LED displays.

CALLED BY : MAIN, FIRSET, RXCHAR, RERR

CALLS ROUTINES : MSTRUP

PSEUDO CODE :

```
Save all registers
Get contents of UNR and write to unit no. LED's
IF UNR = 1 THEN
 CALL Mstrup
ELSE
  Get channel number from proper position in UBUFF
  IF Channel no. = 'X' THEN
    Channel no. = 1
  ENDIF
  Or in channel number with unit number
  Output to slave units
  IF Slave unit is not there THEN
    Channel number = 'X'
    Output "AB" Hex to Channel display (to blank out)
  ELSE
    Strobe in channel number to slave unit
    Output BCD Channel number to display
  ENDIF
ENDIF
Restore registers
```

```
PROCEDURE MSTRUP

OVERVIEW :
     This routine is used to update the master unit relays.

INPUTS :  Channel number in UBUFF

OUTPUTS : Relay position and channel display

CALLED BY : UPDATE

CALLS ROUTINES : None

PSEUDO CODE :

     Clear both Master Register locations
     Set Master Register Pointer to LS byte
     Temp = Channel Number
     IF Channel No. > 8 THEN
       Temp = Channel No. - 8
       Set Master Register Pointer to MS byte
     ENDIF
     Set New_temp = 80H
     For I = 1 to Temp
       Rotate New_temp right one bit
     NEXT I
     Store New_temp at Location pointed at by Master Register Pointer
     Ouptut the two Master Register locations
     Output the channel number to the display
```

PROCEDURE RERR

OVERVIEW :
This routine places a decimal code in the unit and channel LED displays whenever a command error occurs with the remote interface. The code represents the decimal equivalent of the first ASCII letter after the "SW" in te command string.

INPUTS : Output character

OUTPUTS : LED displays

CALLED BY : MAIN

CALLS ROUTINES : UPDATE, ASCBCD

PSEUDO CODE :

```
IF Timer flag (Rtimer) does not equal zero THEN
  Decrement timer flag
  IF timer flag = 0 THEN
    CALL UPDATE
  ELSE
    Get character to display
    Convert to 3 digit BCD
    Output to unit and channel displays
  ENDIF
ENDIF
```

PROCEDURE NEWCHK

OVERVIEW :
This routine checks to see if any units have been added to the system. If units have been added, the module sets the channel to one. If a unit has been removed, the program updates the UBUFF with an "X" in the missing unit channel value.

INPUTS : None

OUTPUTS : None

CALLED BY : MAIN, RXCHAR

CALLS ROUTINES : None

PSEUDO CODE

```
FOR units 2 through 9 and A, B, C, D, E, F, and 0
  Check to see if unit is present
  IF Unit has just been added THEN
    Strobe in channel 1 for the unit
  ENDIF
  IF Unit has just been removed THEN
    Replace channel number in UBUFF with an "X"
  ENDIF
NEXT Unit
```

```
PROCEDURE RXCHAR

OVERVIEW :
     This routine accepts new input characters from the remote
     device and when a termination character is received, it decodes
     and performs the appropriate task.

INPUTS :  New character in register 'B'

OUTPUTS : Updates units and queues message requests for remote devices

CALLED BY :  RINTR, GINTR

CALLS ROUTINES : UPDATE, COMPAR, STCHN1, NEWCHK, HEXBCD

PSEUDO CODE :

     IF Number of characters in receive buffer = 7 THEN
      Set overflow flag
     ENDIF
     IF Character is not a terminator THEN
       IF Overflow flag is not set THEN
         Store character in next spot in receive buffer
       ENDIF
     ELSE
       IF First two letters in string = "SW" THEN
         IF No overflow condition exists THEN
           IF Command = "RE" THEN
             Set remote flag
           ELSE
             IF Remote flag enabled THEN
               CASE Command OF
                 CASE (Command = "RD")
                   Reset remote enable flag
                 CASE (Command = "GL")
                   Reset remote enable flag
                 CASE (Command = "E1")
                   Set echo enable flag
                 CASE (Command = "E2")
                   Reset echo enable flag
                 CASE (Command = "WY")
                   Get pointer to "who you" buffer
                   CALL Queup
                 CASE (Command = "QS")
                   IF Status buffer is free THEN
                     Get status register
                     Add ascii"A" to it
                     Put in status buffer and get pointer
                     CALL Queup
                   ENDIF
                 CASE (Command = "QC")
                   CALL Newchk
                   IF Configuration buffer is free THEN
                     FOR Units 1 to 8
                       Convert channel to ASCII
                       Place in buffer
                       Add a space
                     NEXT Unit
                   IF Any unit 9,A,B,C,D,E,F,0 are present THEN
                     Add CR and LF to buffer
                     FOR Units 9 to 0
                       Convert channel to ASCII
                       Place in buffer
```

```
                    Add a space
                  NEXT Unit
                ENDIF
                Add a ETX, CR and LF to buffer
                Get pointer to buffer
                CALL Queup
              CASE (Command = "CL")
                CALL Stchn1
            END_CASE
          ELSE
            Set command error in status register
            Save third letter in buffer for LED display
            Set timer flag, Rtimer
          ENDIF
        ENDIF
      ELSE
        IF Remote enabled THEN
          IF Recieve buffer has 6 characters , 3rd char.="U",
                4th char=valid unit no. , and 5th and 6th char =
                valid channel no. THEN
            Decode unit number and channel number
            Update UNR and UBUFF
            CALL Update
            IF Unit number did not exist THEN
              Set bit in status register
            ENDIF
          ELSE
            Set command error in status register
            Save third letter in buffer for LED error display
            Set timer flag, Rtimer
          ENDIF
        ENDIF
      ENDIF
    ELSE
      IF Remote enabled THEN
        Set command error in status register
        Save third letter in buffer for LED error display
        Set timer flag, Rtimer
      ENDIF
    ENDIF
    Reset buffer
  ENDIF
```

```
PROCEDURE  RINTR

OVERVIEW :
     This routine is called when an interrupt occurs and the RS-232
     interface is present.  It handles both receiver and transmitter
     ready interrupts and takes the appropriate action.

INPUTS :  Character from remote device

OUTPUTS : Character to remote device

CALLED BY :  INTR

CALLS ROUTINES :  GETBYT,RXCHAR

PSEUDO CODE

     IF Error bit is set THEN
       Reset error condition
       Clear character
     ELSE
       IF Receiver ready THEN
         Input character
         IF Character is XOFF THEN
           Disable Transmit flag
         ELSEIF Character is XON THEN
           Enable transmit flag
         ELSEIF Character is a null or rubout
           Exit
         ENDIF
         Store character
         CALL Rxchar
         IF Transmit flag enabled and echo enabled
           Set Echo in progress flag
         ENDIF
     ENDIF
     IF Output in progress or echo in progress or transmitter turned on THEN
       IF Transmitter not enabled THEN
         Turn off transmitter
         Reset echo in progress flag
       ELSEIF Transmitter ready
         IF Null counter is non-zero THEN
           Send a null
           Decrement null counter
           Clear Echo in progress
           Turn transmitter on
         ELSEIF  Echo in progress THEN
           Output echo character
           Reset Echo flag
           Turn on transmitter
         ELSEIF  Output in progress THEN
           CALL Getbyt
           IF Character = 0 THEN
             Turn off transmitter
             Clear output in progress flag
           ELSEIF  Character = CR
             Output character
             Set null counter
           ELSE
             Output character
             Reset echo flag
             Turn on transmitter
```

```
      ENDIF
    ELSE
      Turn off transmitter
    ENDIF
  ENDIF
ENDIF
```

```
PROCEDURE GINTR

OVERVIEW :
     This routine is called when an interrupt occurs and the GPIB
     Interface is present.  It decodes the cause of the interrupt and
     takes appropriate action.

INPUTS : Characters from remote device

OUTPUTS : Characters to remote device

CALLED BY : INTR

CALLS ROUTINES :  RXCHAR, GETBYT, STCHN1

PSEUDO CODE

     IF Interrupt occurred from the GPIB interface THEN
       CASE Interrupt type OF
         CASE (Interrupt = command)
           IF Remote/local change THEN
             IF Remote enabled THEN
               Set flag
               Turn on LED
             ELSE
               Reset flag
               Turn off LED
             ENDIF
           ELSEIF  Device Clear THEN
             CALL Stchn1
           ELSEIF  Unrecognized Command THEN
             Release Dac
           ENDIF
         CASE (Interrupt = byte out)
           IF Output in progress THEN
             CALL Getbyt
             IF Last character THEN
               Reset output in progress flag
               Disable BO interrupt
               Set EOI bit
             ENDIF
             IF Character is non-zero THEN
               Output character
             ENDIF
           ENDIF
         CASE ( Interrupt = byte in )
           Input character
           CALL Rxchar
           IF EOI is set THEN
             Fetch a line feed
             CALL Rxchar
           ENDIF
       END_CASE
     ENDIF
```

PROCEDURE GETBYT

OVERVIEW :
This routine extracts the next character that should be sent to the remote device. A circular queue of message pointers is accessed to get the current output message. An index is then applied to that message to extract the correct byte. If a zero is accessed, the routine goes to the next message. IF there are no messages, a zero is returned.

INPUTS : None

OUTPUTS : Character to output or zero.

CALLED BY : GINTR, RINTR

CALLS ROUTINES : None

PSEUDO CODE :

```
IF Queue is empty THEN
  Set returned byte to zero
ELSE
  Fetch output index into circular queue
  Fetch current output message pointer
  Add current offset into message
  Fetch character
  IF Character = 0 THEN
    Free buffer in use
    Clear offset
    Increment output index
    Make circular adjustment
    Decrement number of entries
    Process next character (GOTO Getbyt)
  ELSE
    IF next character = 0 THEN
      Set flag to indicate last character (EOI)
    ENDIF
    Increment offset
  ENDIF
ENDIF
```

PROCEDURE QUEUP

OVERVIEW :
This routine is used to place pointers of desired output messages in a circular queue.

INPUTS : Pointer to message

OUTPUTS : None

CALLED BY : RXCHAR

CALLS ROUTINES : None

PSEUDO CODE :

```
IF Circular Queue is not full THEN
  Increment number of entries
  Place pointer on queue at current input index
  Increment input index
  Make circular adjustments
  IF IEEE-488 Interface present THEN
    Enable "BO" Interrupt
  ELSEIF RS-232 Interface present THEN
    Enable "TX" Interrupt
  ENDIF
ENDIF
```

APPENDIX II

DATA BASE DESCRIPTION

Data Base Dictionary

"UNR" - Unit number register. It contains the current unit number that appears in the display. This variable is passed to UPDATE when a change takes place.

"UBUFF" - This is a 16 byte array that contains the channel numbers for each unit. UBUFF (1) = Channel for unit 1. UBUFF (16) = Channel no. for unit 0.

"CHAN16" - This varialble is 10 if there are no units in system assigned to unit numbers A,B,C,D,E,F, or 0. Otherwise it is set to 17. It is used in MAIN to determine if a rollover from unit 9 to unit 1 should take place. It is continually updated by NEWCHK.

"RXBUFF" - This an 8 byte array to hold the remote input characters. The first byte indicates the count of received byte since the last terminator.

"OVRFLO" - This is a flag that indicates that the received string is too long for any valid command.

"TSBUFF" - Buffer used to format the status query reply.

"TXBUFF" - Buffer used to format the configuration query reply.

"WYBUFF" - Fixed message in ROM for " Who are you" query. This string is in same file as RXHCAR.

"CIRCLB" - Circular queue to hold message pointers that are to be sent out to the remote device. It has capacity to hold up to 8 message pointers

"POINT" - Byte pointer into current output message

"NUMMES" - Number of entries in the circular queue.

"ININD" - Input index into CIRCLB

"OUTIND" - Output index into CIRCLB

"STATRG" - Status register (power fail, command error, absent box)

"INSTRG" - Holds the most recent reading of the front panel switches

"UNDSRG" - Holds the current value that has been output to the unit display port (68H).

"MSTRG" - Two byte buffer that holds the current value that has been output to the master relay control ports (18H and 10H)

"CRCSAV" - Two byte buffer that holds the CRC result.

"NCOUNT" - Null counter for RS-232 after a carriage return.

"RCDBYT" - Received RS-232 character that is to be echoed.

"SAVPT" - Temporary two byte location to hold message pointer in subroutine GETBYT.

"ECHOFL" - Echo enable flag

"REMEN" - Remote enable flag

"LAMPTS" - Lamp test delay timer.

"OUTIP" - Remote output in progress flag

"TRON" - Flag that indicates the current state of transmit enable on the 8251.

"TRANSM" - Flag to indicate XON/XOFF status.

"LAST" - This flag is set on the last character of an output message.

"EOISET" - This flag is used to indicate whether EOI was set on an input from the 68488 GPIB Interface.

"RSTHER" - Flag to indicate if the RS-232 interface is present in this configuration.

"GBTHER" - Flag to indicate if the GPIB interface is present in this configuration.

"EIP" - Echo in progress flag

"RPTFL" - Auto repeat timer for front panel switches

"FLAG1" - General flag used in RXCHAR.

"BADFLG" - Flag that is set by UPDATE when the unit given is not present.

"RTIMER" - Counter that is used to time command error characters on the LED displays.

"BCHAR" - Command error character